Then, only if no matches are found, will FileMan make a second pass through the indexes looking for partial matches.

U Normally the lookup value is expected to be in external format (for dates, pointers and such). FileMan first searches the requested index for a match to the user input as it was typed in. Then, if no match is found, FileMan automatically tries certain transforms on the lookup value.

For instance, if one of the lookup indexes is on a date field, FileMan tries to transform the lookup value to an internal date, then checks the index again. The U flag causes FileMan to look for an exact match on the index and to skip any transforms. Thus the lookup value must be in internal format. This is especially useful for lookups on indexed pointer fields, where the internal entry number (i.e., internal pointer value) from the pointed-to file is already known.

Ordinarily this flag would not be used along with the "A", "B", "M", "N" or "T" flags. In many cases it makes sense to combine this with the "X" flag.

V If DIC(0) contains V and only one match is made to the user's lookup value, then they will be asked "OK?" and they will have to verify that the looked-up entry is the one they wanted. This is an on the fly way of getting behavior similar to the permanent flag that can be set on a file by answering "YES" to the question "ASK 'OK' WHEN LOOKING UP AN ENTRY?" (See the EDIT FILE option within the FileMan UTILITY option, described in the Advanced User Manual).

X If DIC(0) contains X, for an exact match, the input value must be found exactly as it was entered. Otherwise, the routine will look for any entries that begin with the input X. Unless 'X-act match' is specified, lowercase input that fails in the lookup will automatically be converted to uppercase, for a second lookup attempt. The difference between X and O (described above) is that X requires an exact match. If there is not one, either DIC exits or tries to add a new entry. With O, if there is not an exact match, DIC looks for a partial match beginning with the input.

Z If DIC(0) contains Z and if the lookup is successful, then the variable Y(0) will also be returned. It will be set equal to the entire zero node of the entry that has been found. Another

array element, Y(0,0), is also returned and will be set equal to the printable expression of the .01 field of the entry selected. This has no use for Free Text and Numeric data types unless there is an OUTPUT transform. However, for Date/Time, Set of Codes and Pointer data types, Y(0,0) will contain the external format.

Adding New Subentries to a Multiple

You can use ^DIC or FILE^DICN to add new subentries to a multiple. In order to add a subentry, the following variables need to be defined:

DIC — Set to the full global root of the subentry. For example, if the multiple is one level below the top file level: file's_root,entry#,multiple_field's_node,

DIC(0) — Must contain "L" to allow LAYGO.

DIC("P") — Set to the 2nd piece of 0-node of the multiple field's DD entry. **NOTE:** As of Version 22 of FileMan, the developer is no longer required to set DIC("P"). The only exception to this is for a few files that are not structured like a normal FileMan file, where the first subscript of the data is variable in order to allow several different 'globals' to use the same DD. An example of this is the FileMan Audit files where the first subscript is the file number of the file being audited.

DA(1)...

DA(n) — Set up this array such that DA(1) is the IEN at the next higher file level above the multiple that the lookup is being performed in, DA(2) is the IEN at the next higher file level (if any), ... DA(n) is the IEN at the file's top level.

NOTE: The value of the unsubscripted DA node should not be defined when doing lookups in a subfile—that's the value you're trying to obtain!

Below is an example of code that:

1. Uses ^DIC to interactively select a top-level record.
2. Uses ^DIC to select or create a **subentry** in a multiple in that record.
3. Uses ^DIE to edit fields in the selected or created subentry.